MULTIMEDIA UNIVERSITY®

STUDENT ID NO

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 2, 2019/2020

HPB3021 –PARALLEL COMPUTING

(All sections / Groups)

5 MARCH 2020
9:00 - 11:00 AM
(2 hours)

INSTRUCTIONS TO STUDENTS

1. This question paper consists of 5 pages, including this cover page.
2. You are required to attempt all questions. All questions carry equal marks (10).
3. Write all your answers in the Answer Booklet provided.
4. You may use a calculator.

Question 1

a) A bioinformatician downloaded source file for a comparative genomic application program. The rules are the program can be read by all, executed by user and group and written by user only. Write out the command for the stated rules. [2 marks]

b) What actions can we achieve from the *top* command? List any two. [2 marks]

c) What is the function of symbolic link? What is the command used to create the symbolic link? [1 mark]

d) A student performed preprocessing on the fastq formatted sequencing reads. Given the identifier of the reads is "@", how does the student count the number of reads after the preprocessing? [1 mark]

e) What is an inode? [1 mark]

f) After executing a program, many temporary files were generated in the working directory. Write the command to delete all the ".tmp" files then move all the output files to the home directory of the user, in single command line. [2 marks]

g) Suggest any one location to install application that can be executed by all users in Linux. [1 mark]

Question 2

a) What are the four classes of communication in MPI? [2 marks]

b) What are the four criteria to evaluate a switch network? [2 marks]

c) Given binary tree network with 7 levels, specify:
i. the total number of nodes in the network
ii. the level when the number of node is 16 [2 marks]

d) What are the two partitioning methods in Foster's methodology? Explain the differences. [4 marks]

Question 3

a) A non-sudoer locally installed Velvet assembler in the home directory. How can the user execute the assembler without specifying the full path or running from the Velvet directory? Specify the command. [2 marks]

Continued .....

b) Being a comparative genomics expert, one should expect to execute long process computational analyses. How does user keep the execution on going without interruption, even though the commands are remotely executed? List any ONE command for the purpose. [1 mark]

c) What are the 4 design steps in Foster's Design Methodology? [2 marks]

d) Imagine we have the following block of code that is meant to be run on parallel computers. Draw a data dependence graph to perform the parallelism on the given code. Identify different types of parallelism on the graph and justify what is the optimal number of computers to process the code with a simple justification. [5 marks]

```
- A = 2
- B = A + 1
- C = A + 2
- D = 0
- While (D <= 500)
-     D = B * C
```

Question 4

a) The following program calculates the sum of the elements of an array using MPI parallelism. The root process acts as a master and sends a portion of the array to each child process. Master and child processes then all calculate a partial sum of the portion of the array assigned to them, and the child processes send their partial sums to the master, who calculates a grand total. Complete the parts labelled with A-K in the following program: [10 marks]

```
#include <stdio.h>
 //include library
```

A

```
   #define max_rows 100000
   #define send_data_tag 2001
   #define return_data_tag 2002

   int array[max_rows];
   int array2[max_rows];

   void main(int argc, char **argv)
   {
      long int sum, partial_sum;
      MPI_Status status;
      int my_id, root_process, ierr, i, num_rows, num_procs,
         an_id, num_rows_to_receive, avg_rows_per_process,
         sender, num_rows_received, start_row, end_row,
num_rows_to_send;
      root_process = 0;
```

```
//initiate MPI
```

B

```
   // find out MY process ID, and how many processes were started
```

C

D

```
      if(my_id == root_process) {


         printf("please enter the number of numbers to sum: ");
         scanf("%i", &num_rows);

         if(num_rows > max_rows) {
            printf("Too many numbers.\n");
            exit(1);
         }

         avg_rows_per_process = num_rows / num_procs;


         for(i = 0; i < num_rows; i++) {
            array[i] = i + 1;
         }


         for(an_id = 1; an_id < num_procs; an_id++) {
            start_row = an_id*avg_rows_per_process + 1;
            end_row   = (an_id + 1)*avg_rows_per_process;

            if((num_rows - end_row) < avg_rows_per_process)
               end_row = num_rows - 1;

            num_rows_to_send = end_row - start_row + 1;

//process 0 sends out "num_rows_to_send" and "array[start_row]" to
"an_id" //processes


```

E

F

```
         }

         /* and calculate the sum of the values in the segment
assigned
          * to the root process */

         sum = 0;
         for(i = 0; i < avg_rows_per_process + 1; i++) {
            sum += array[i];
         }

         printf("sum %i calculated by root process\n", sum);

         /* and, finally, I collet the partial sums from the slave
processes,
```

```
          * print them, and add them to the grand sum, and print it
*/
         for(an_id = 1; an_id < num_procs; an_id++) {
// A receive function of "partial_sum" from any sources
```

G

```
            sender = status.MPI_SOURCE;
            printf("Partial sum %i returned from process %i\n",
partial_sum, sender);
            sum += partial_sum;
         }
         printf("The grand total is: %i\n", sum);
      } //end of if process is 0
      else { //other processes
         //Receive "num_of_rows" and array2 from root process
```

H
I

```
         num_rows_received = num_rows_to_receive;
         partial_sum = 0;
         for(i = 0; i < num_rows_received; i++) {
            partial_sum += array2[i];
         }
// A send function of "partial_sum" to root process
```

J

```
      }
// close down MPI
```

K

```
   }
```

END OF PAPER